3D Visualization of Truecolor Image Histograms

Paula J. REITAN
Department of Computer Science & Electrical Engineering
University of Maryland Baltimore County
1000 Hilltop Circle
Baltimore, Maryland 21250
UNITED STATES
reitan@scs.usna.navy.mil
http://www.scs.usna.navy.mil/~reitan

Truecolor Images

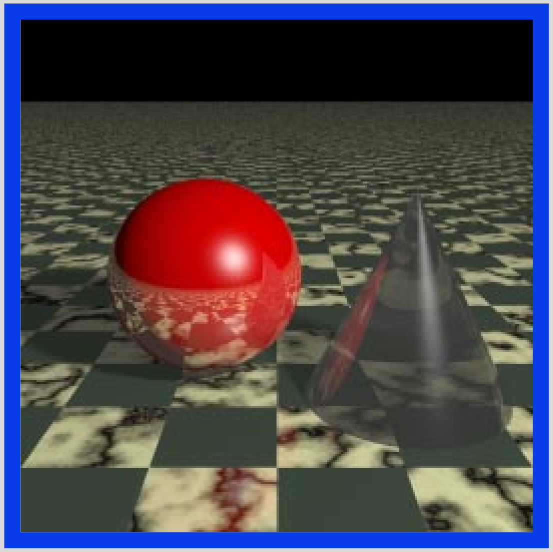
- A ***truecolor image***

  $$f\colon \mathbb{Z}_W \times \mathbb{Z}_H \rightarrow C,$$

  where $f(x, y)$ is the RGB color of the pixel at column x, row y of image f and

  $$C = \{\, \mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N \,\} \subseteq \mathrm{RGB}.$$

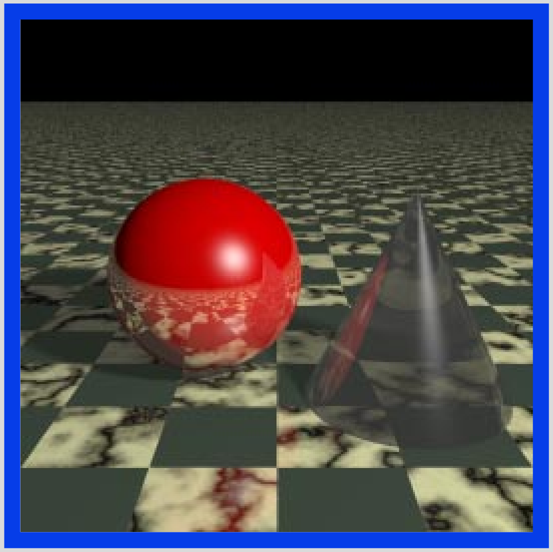
- $\mathrm{RGB} = \{\, (r, g, b) \mid r, g, b \in \mathbb{Z}_{256} \,\}.$



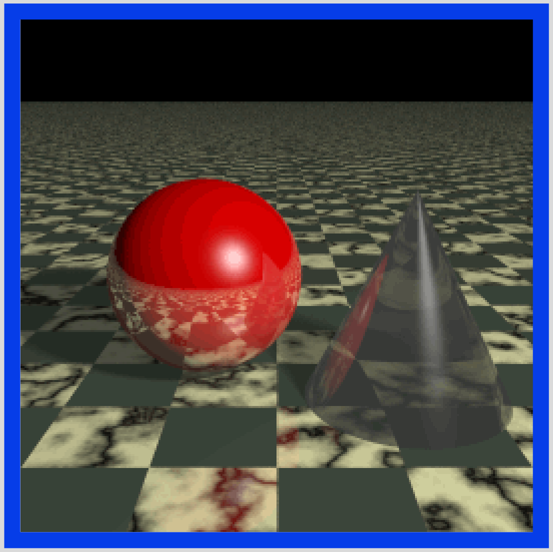
24. Solids,
$W = 512$, $H = 512$,
$N = 22{,}895$

Color Image Quantization

- *The irreversible* transformation of a truecolor image into a *color-mapped* image consisting of *K* carefully selected representative colors.



Truecolor image
N = 22,895



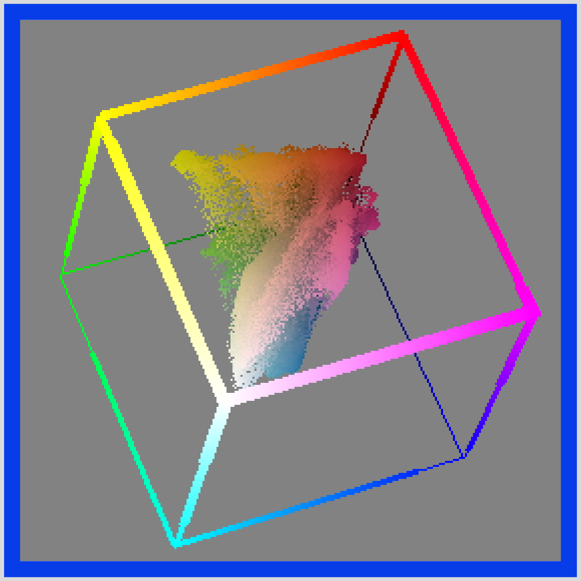
Color-mapped image
K = 256

Truecolor Image Histograms

- The first step of many color quantization techniques is to create a histogram of the colors in the truecolor image.
- The ***histogram***

$$\boldsymbol{H}(f)\colon \mathrm{RGB} \rightarrow \mathbb{N},$$

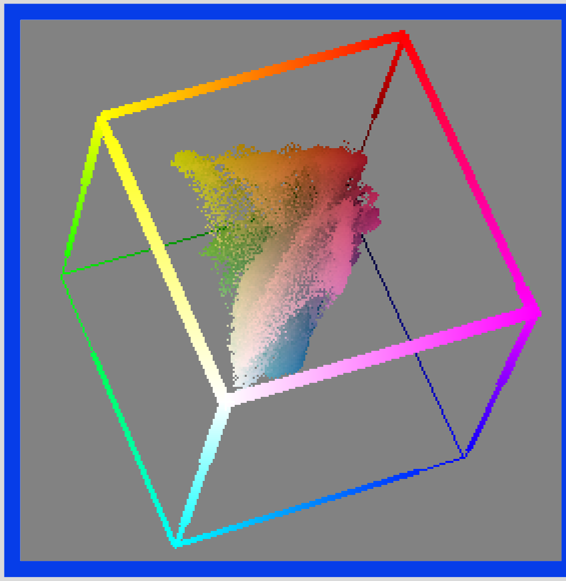
where $H_f(\boldsymbol{c})$ is the number of pixels in the truecolor image f with color $\boldsymbol{c}$.



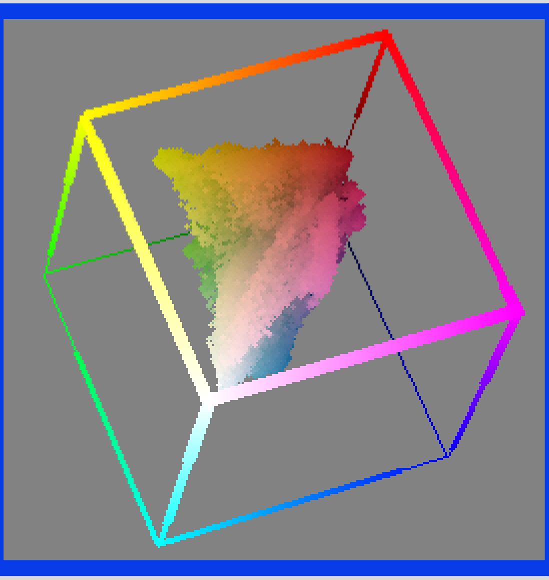
10. Windsails,
$W = 768$, $H = 512$,
$N = 34{,}111$

Bit-cutting

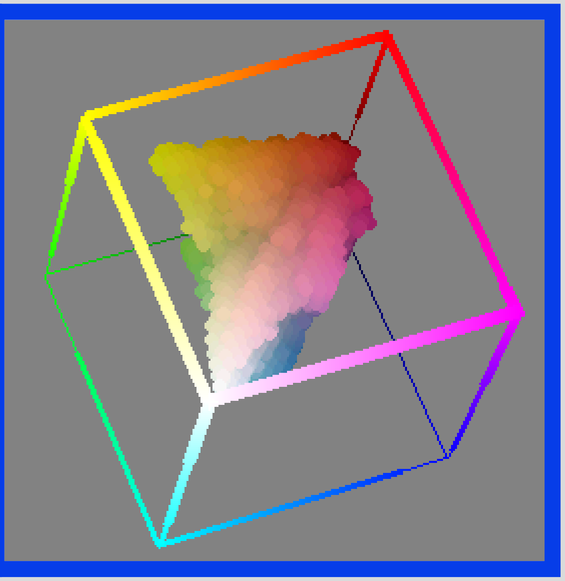
- By cutting the (8-*p*) least significant bits from each *r*, *g*, *b* component, the RGB color space is uniformly quantized to a smaller color space denoted as RGB_p.
- The number of unique RGB_p colors in a truecolor image will be denoted as N_p.



RGB_7: $N_7 = 31{,}810$

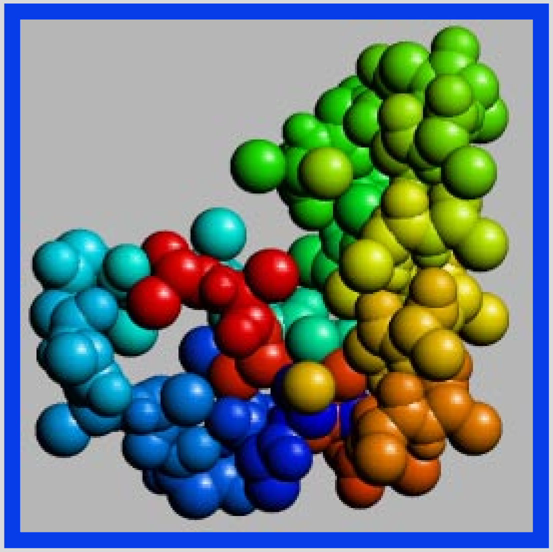


RGB_6: $N_6 = 12{,}100$

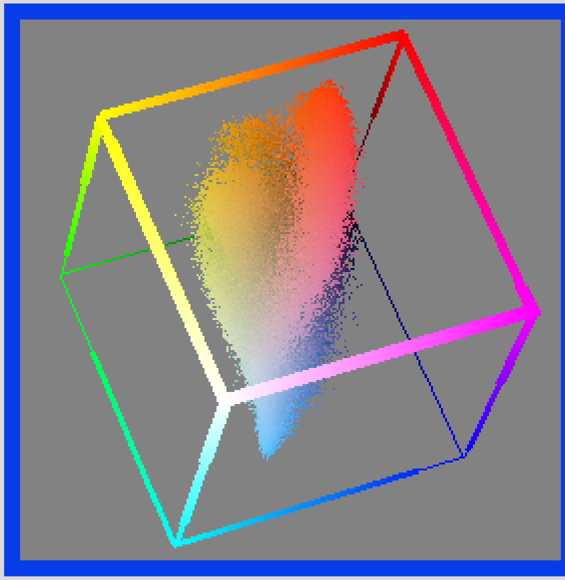


RGB_5: $N_5 = 2{,}978$

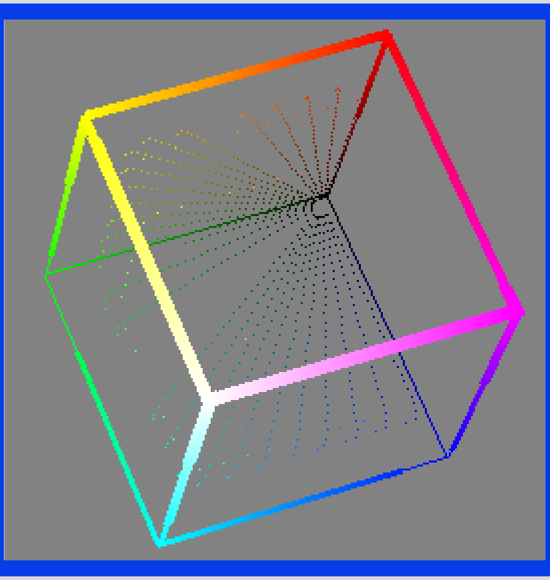
3. Mandrill



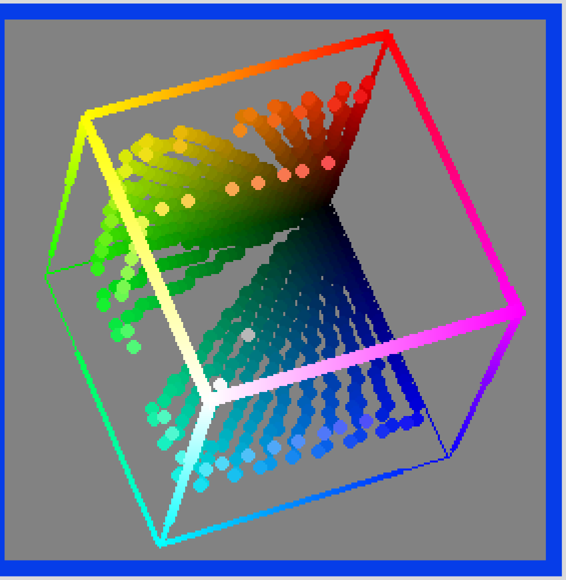
27. Crambin



$N_8 = 216{,}200$

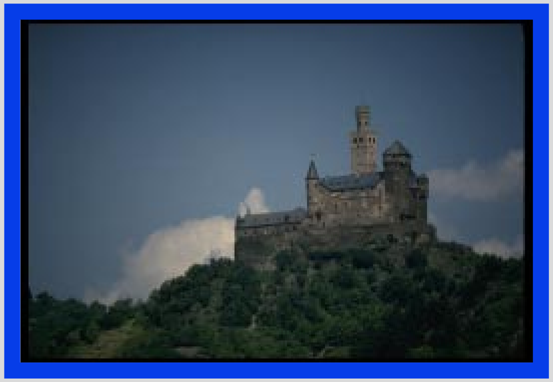


$N_8 = 856$

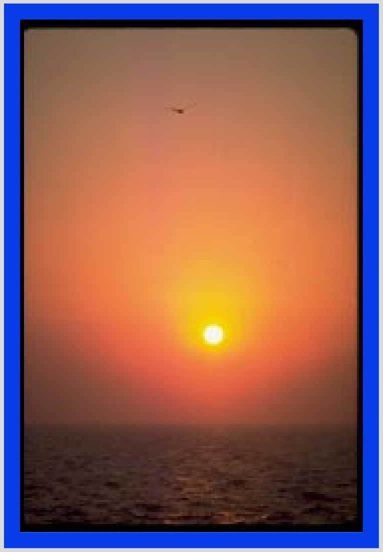


🟪 🟪 🟦 🟦 🟩 🟨 🟧 🟥

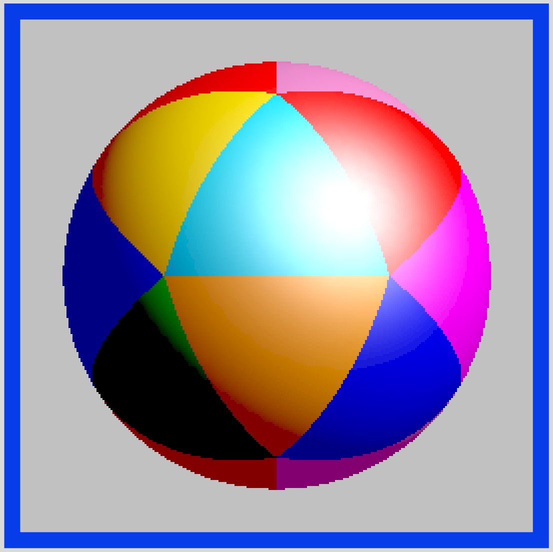
$N_5 = 748$



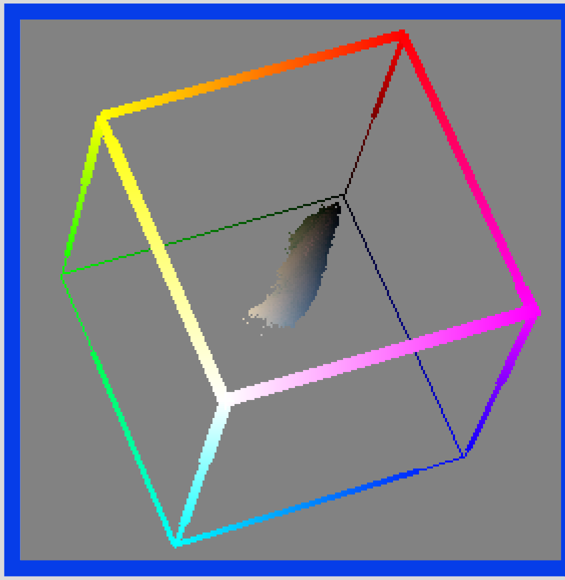
18. Castle



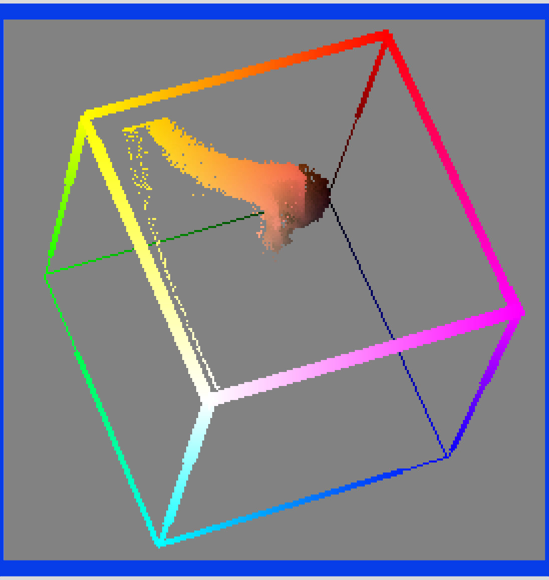
20. Sunset2



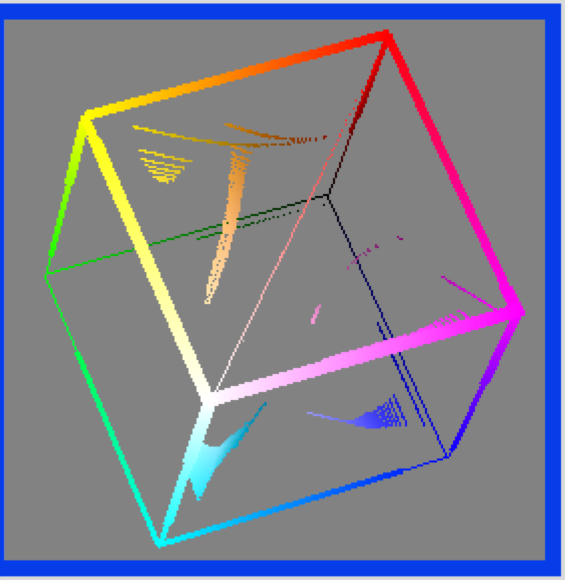
23. Beachball



$N_8 = 11{,}149$



$N_8 = 71{,}687$



$N_8 = 4{,}969$

Histogram Performace Analysis

- We are in search of a **space efficient** 3D histogram data structure that also supports **fast** insertion of pixels, query for H_f, and rendering.
- The following performance measures are of interest:
  - ***Space utililzation***
  - ***Insertion Time*** (T_I/t_I)
  - ***Query Time*** (T_Q/t_Q)
  - ***Build Time*** (T_B/t_B)
  - ***Render Time*** (T_R/t_R)
  - ***Height*** (h)

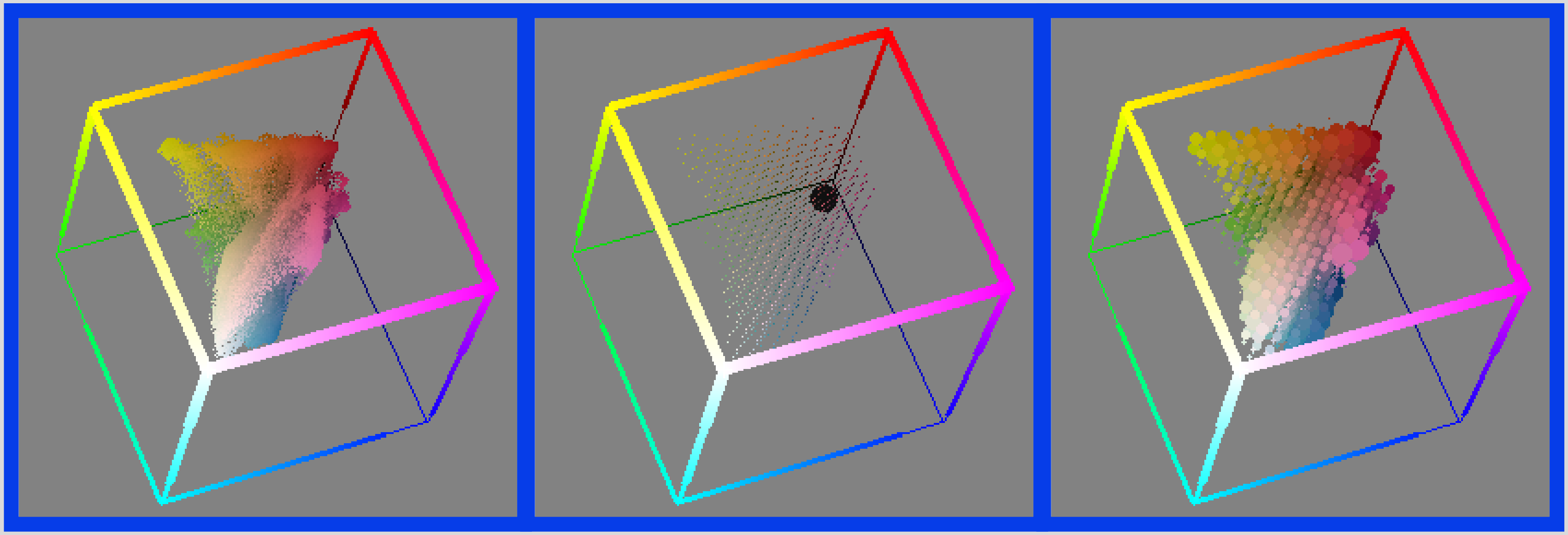
3D Arrays

- A 3D array of $2^{p\text{x}}2^{p\text{x}}2^{p}$ natural numbers used to store H_f.
- ***Advantages:*** $T_I = \Theta(1)$ and $T_Q = \Theta(W \cdot H)$.
- ***Disadvantages:*** low space utilization and high render times.

	RGB_8			RGB_5		
	λ	t_B	t_Q	λ	t_B	t_Q
Max	0.0129	9.120	4.540	0.2576	4.700	4.360
Min	0.0001	4.730	0.740	0.0150	0.780	0.720
Avg	0.0030	8.132	3.748	0.0809	3.925	3.643

3D Arrays

- Excellent choice for $\text{RGB}_{p}\leq_{5}$.
- We need a dynamic data structure whose space utilization and render time scales better with respect to *p*.



$p = 8$, $t_R = 18.58$, $\lambda = 0.00051$

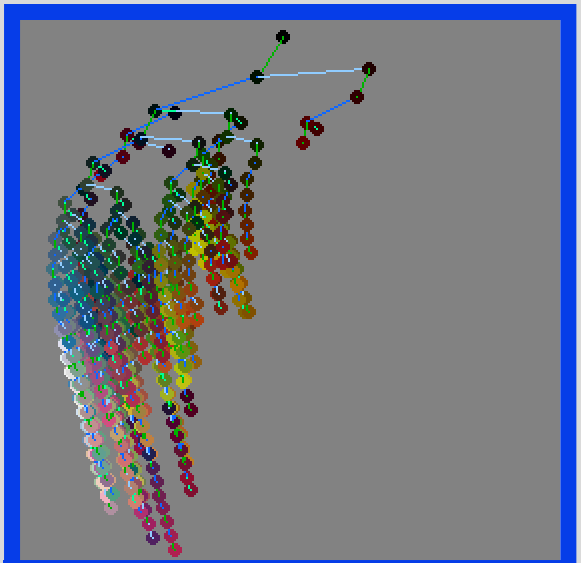
$p = 4$, $t_R = 0.14$, $\lambda = 0.2612$ scaled by *count*

p = 4, $t_R = 0.07$, scaled by log(*count*)

Binary Search Trees

- The nodes of BST histograms are ordered pairs,
  $$(\mathbf{c} \in \mathrm{RGB}_p,\ count \in \mathbb{N}),$$
  where $count = H_f(\mathbf{c})$.
- We convert $\mathbf{c} \in \mathrm{RGB}_p$ into a unique unsigned integer:
  $$\textbf{key}(\ \mathbf{c}\ ) = (\ \mathbf{c}.b \ll 16) \mid (\ \mathbf{c}.g \ll 8) \mid \mathbf{c}.r.$$
- Will be used as bucket structures for the spatial subdivisions techniques discussed later.

Binary Search Trees



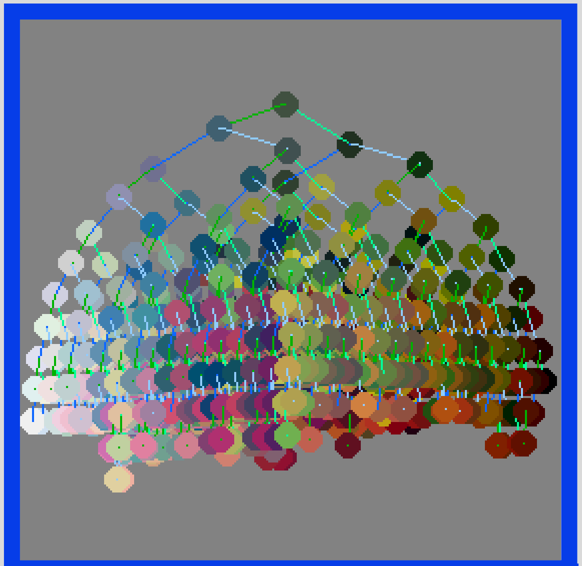
BST

$p = 4$,

$N_4 = 1{,}070$

$h = 31$,

$t_R = 0.22$

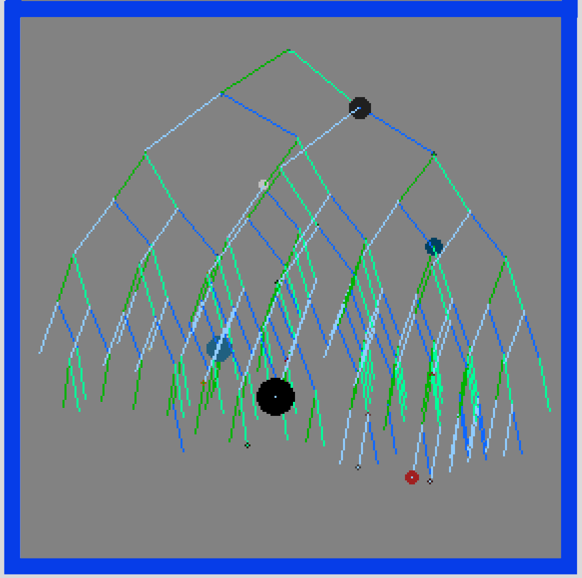


Red-black

$p = 4$,

$h = 12$,

$t_R = 0.35$



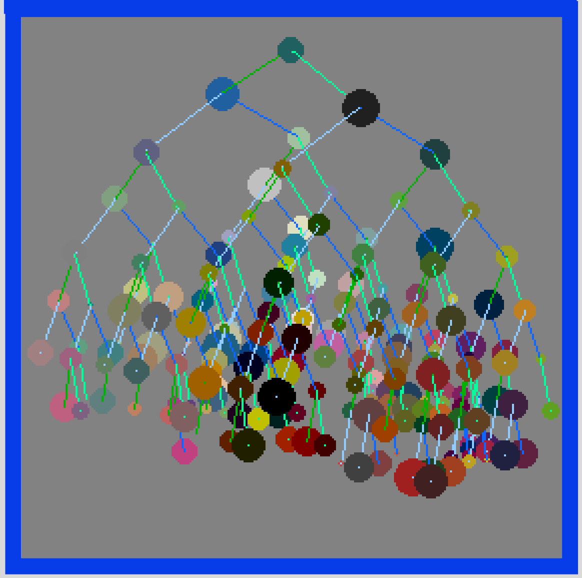
AVL

$p = 3$,

$N_3 = 198$

$h = 8$,

$t_R = 0.04$



AVL

$p = 3$,

$t_R = 0.08$

scaled by *count*

scaled by log(*count*)

Binary Search Trees

- Let T be a BST histogram of a truecolor image. Let d_T be the depth of a node in the BST. We define the structural query time of T:

$$T_Q = \sum_{n \in T} (d_T(n) + 1) \cdot c_T \cdot n.count,$$

where $c_T = 2$ is the number of comparisons made at each node.

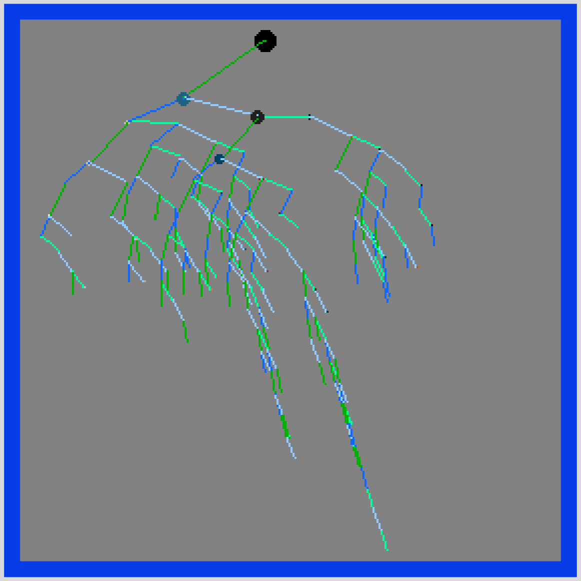
Treaps

- Simultaneously maintain the BST property on

  key(*node.c*)

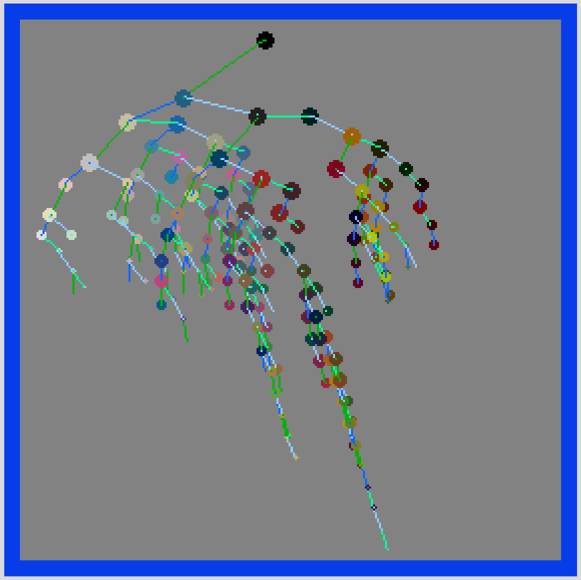
  and the max heap property on

  node.count



$p = 3$,
$h = 21$,
$t_R = 0.03$

scaled by *count*



$p = 3$,
$t_R = 0.04$

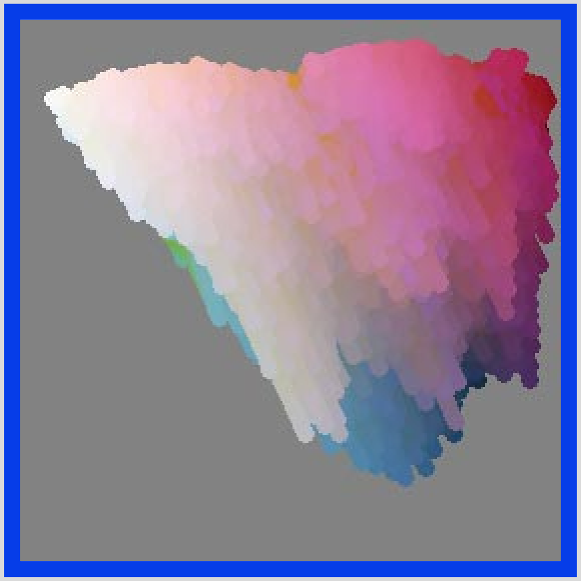
scaled by log(*count*)

Binary Search Trees

		RGB_8			RGB_5		
		h	t_B	t_Q	h	t_B	t_Q
BST	Max	703	55.13	108.44	184	29.59	55.56
	Min	38	5.12	9.55	36	2.19	3.33
	Avg	131	30.61	54.72	54	17.68	29.97
Red-black	Max	21	22.16	29.31	16	16.04	22.60
	Min	11	2.45	3.31	10	1.94	2.70
	Avg	18	15.77	22.23	13	11.29	16.42
AVL	Max	21	34.38	29.249	15	22.75	22.14
	Min	11	3.75	3.76	10	2.81	2.52
	Avg	17	23.96	22.10	12	16.79	16.58
Treap	Max	703	31.87	54.19	66	17.72	27.30
	Min	29	2.10	2.64	18	1.75	2.08
	Avg	93	21.21	31.56	39	12.13	16.49

Spatial Subdivision Methods

- Decomposition of RGB_p into smaller pieces called ***partitions*** or ***cells***.
- ***Space utilization*** $\lambda = O/M$, where M is the number of cells and O is the number of non-empty cells.



2D Array

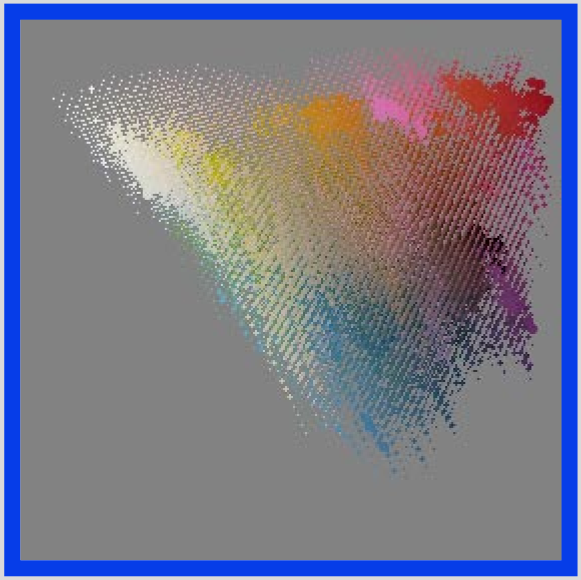
$p = 6$,

$N_4 = 26{,}233$

$M = 4{,}096$,

$\lambda = 0.51$,

$t_R = 3.77$



2D Array

$p = 6$,

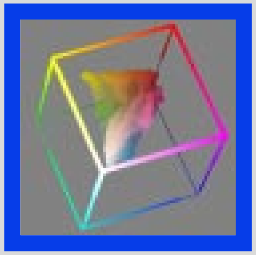
$t_R = 1.28$

scaled by log(*count*) 🟪 🟪 🟦 🟦 🟩 🟨 🟧 🟥

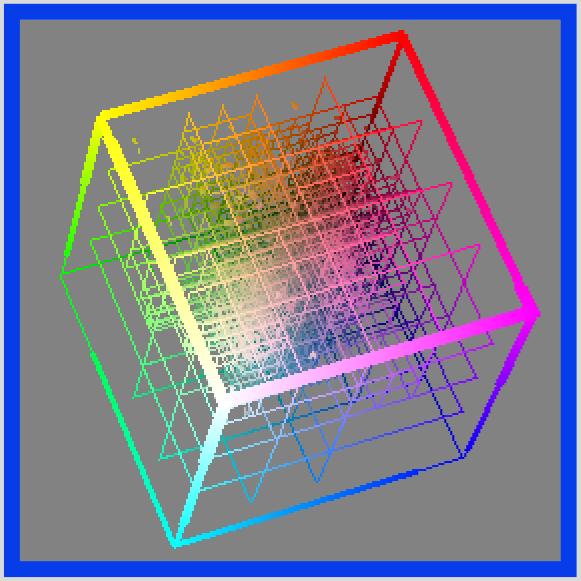
2D Arrays

		RGB_8			RGB_5		
		λ	t_B	t_Q	λ	t_B	t_Q
List	Max	0.433	11.060	10.120	1.000	9.940	9.920
	Min	0.009	1.500	1.170	0.128	1.180	1.140
	Avg	0.228	7.849	7.200	0.448	7.097	7.046
Red-Black	Max	0.433	10.650	9.430	1.000	8.580	9.620
	Min	0.009	1.420	1.150	0.128	1.160	1.170
	Avg	0.228	7.751	7.076	0.448	6.471	7.031
Treap	Max	0.433	12.750	9.220	1.000	10.130	8.610
	Min	0.009	1.660	1.160	0.128	1.450	1.180
	Avg	0.228	9.280	6.925	0.448	7.934	6.553

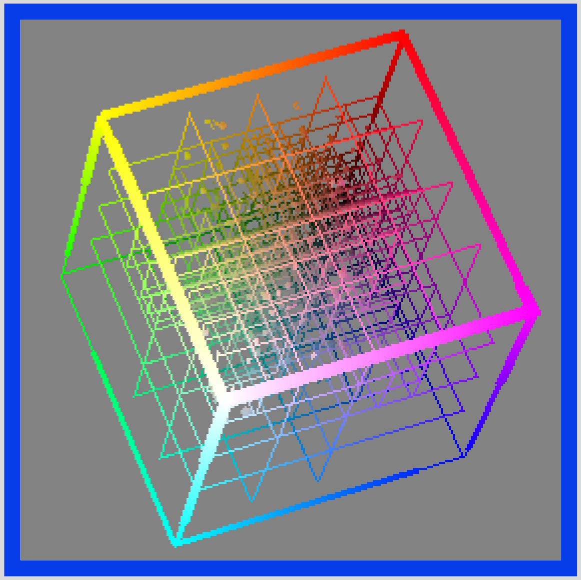
Octrees



- Hierarchical subdivision method.
- When the number of elements in a bucket exceeds the maximum bucket size (*B*), the cell (***octant***) is subdivided into eight pieces using three cut-planes which are orthogonal to each of the R, G, B axes.



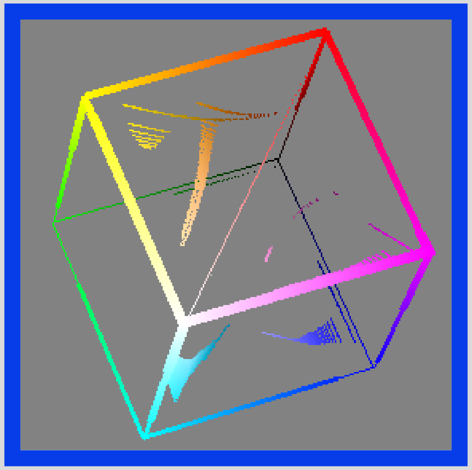
$p = 8$,
$B = 256$,
$N_8 = 86{,}008$
$\lambda = 0.797$,
$t_R = 7.82$

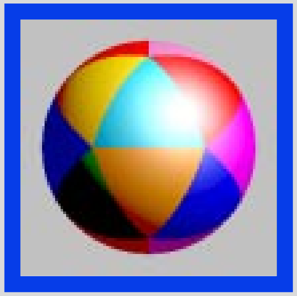


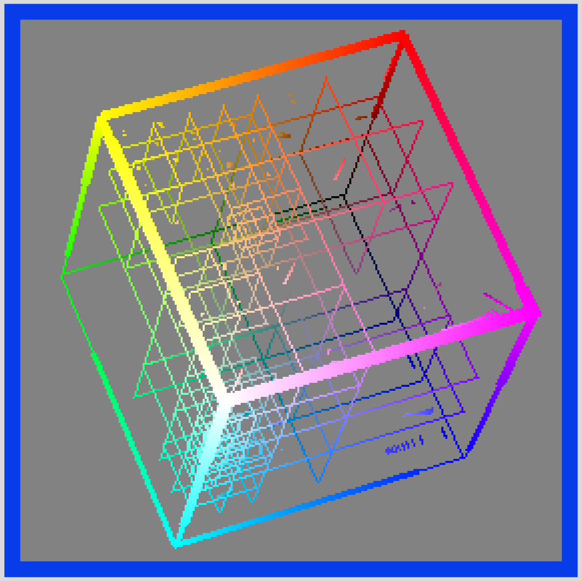
$p = 5$,
$B = 64$,
$N_5 = 5{,}873$
$\lambda = 0.721$,
$t_R = 0.93$

center

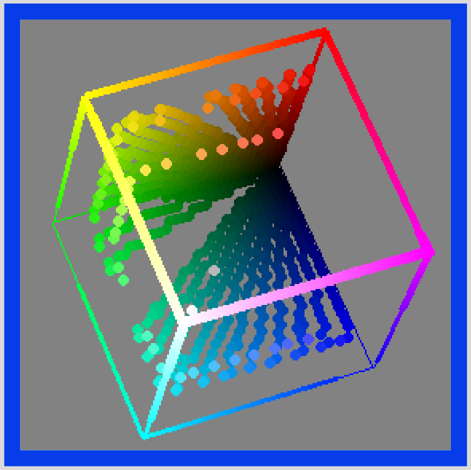
Octrees

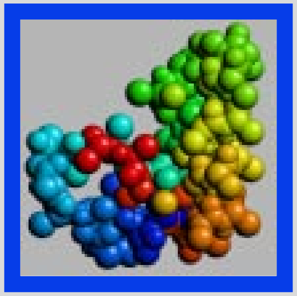


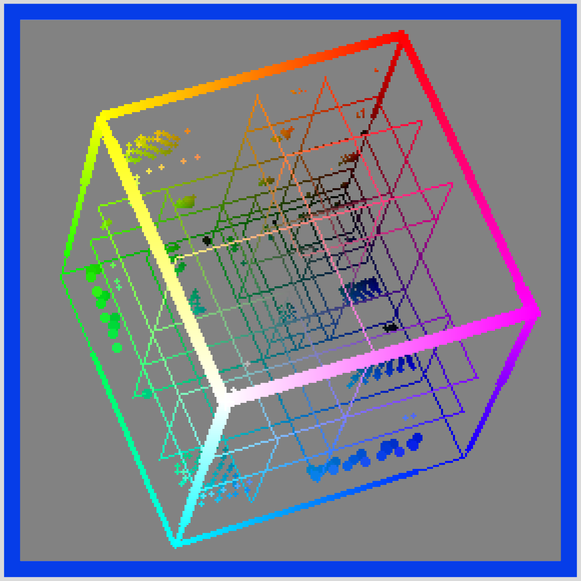




$p = 8$,
$B = 256$,
$N_8 = 4{,}969$
$\lambda = 0.50$,
$t_R = 0.47$







$p = 5$,
$B = 64$,
$N_5 = 637$,
$\lambda = 0.56$,
$t_R = 0.15$

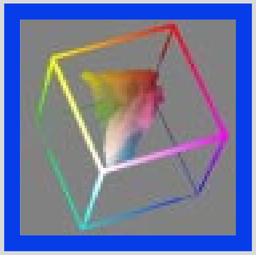
center

Octrees

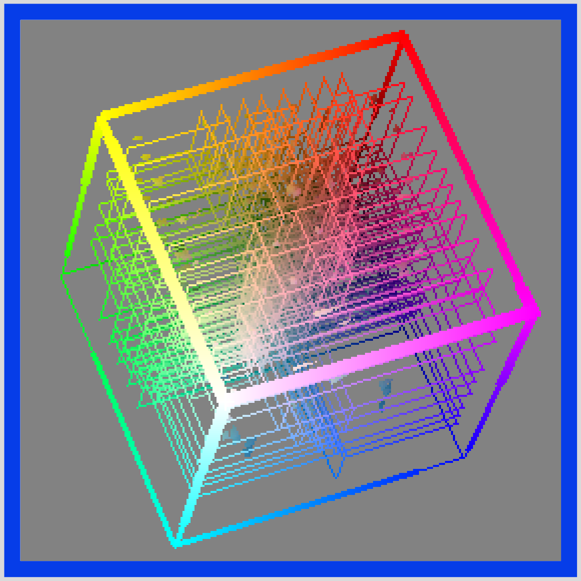
		RGB_8			RGB_5		
		λ	t_B	t_Q	λ	t_B	t_Q
List	Max	0.833	46.40	25.15	0.812	41.04	25.68
	Min	0.416	4.570	2.310	0.421	3.820	2.080
	Avg	0.685	35.36	18.37	0.669	26.24	16.14
Red-black	Max	0.833	29.29	16.45	0.832	18.01	14.71
	Min	0.416	2.800	1.930	0.488	1.990	1.800
	Avg	0.685	20.84	12.84	0.691	13.28	11.32
Treap	Max	0.833	31.20	16.10	0.832	18.39	13.77
	Min	0.416	2.810	1.900	0.488	2.000	1.700
	Avg	0.685	21.74	12.15	0.691	13.33	9.993

B=64, Center

k-d Trees

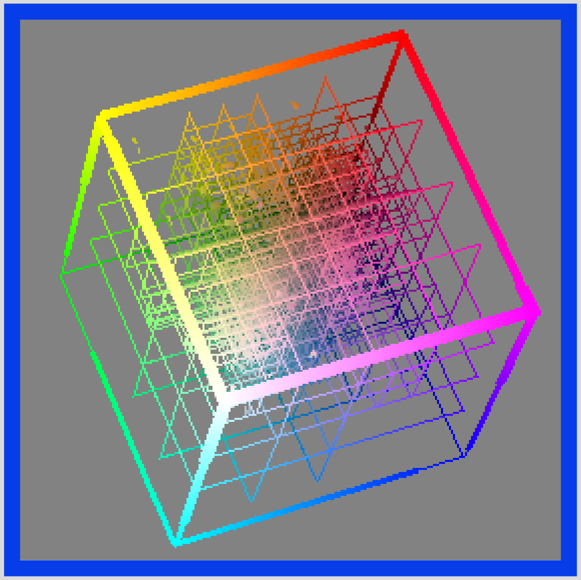


- Hierarchical subdivision method.
- When the number of elements in a bucket exceeds the maximum bucket size (B), the cell subdivided into two pieces using a single cut-plane which is orthogonal to one of the R, G, B axes.



k-d tree
$p = 8$,
$B = 256$,
$N_8 = 86{,}008$
$M = 544$,
$t_R = 6.83$

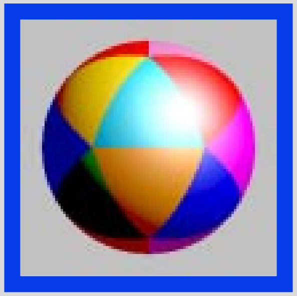
Largest range, center

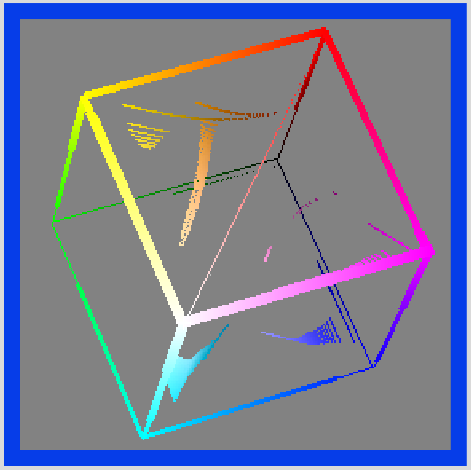


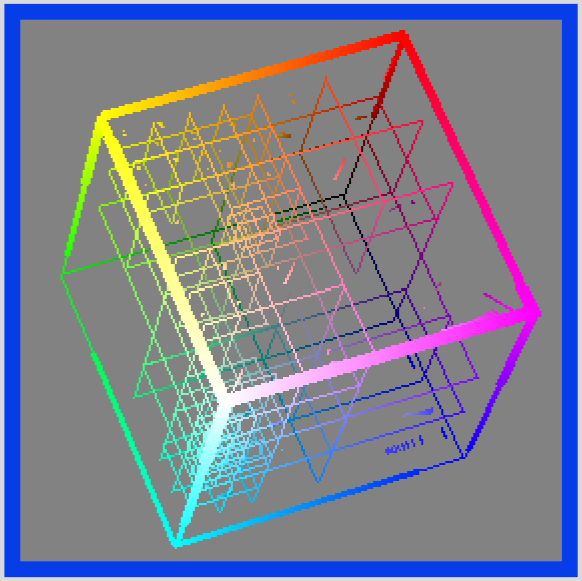
Octree
$p = 8$, 🟪
$B = 256$, 🟪
$M = 1{,}541$ 🟦
$\lambda = 0.797$, 🟦
$t_R = 7.82$ 🟩
🟨
🟧

center 🟪 🟪 🟦 🟦 🟩 🟨 🟧 🟥

k-d Trees







***k*-d tree**

$p = 8,$

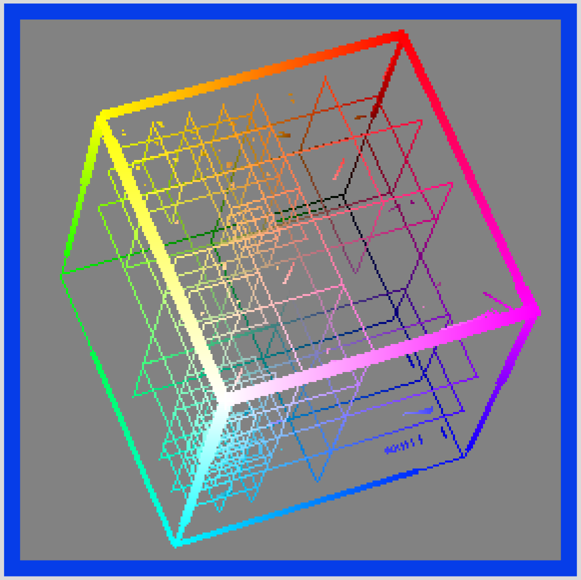
$B = 256,$

$N_8 = 4{,}969$

$M =$

$t_R =$

Largest range, center



Octree

$p = 8,$

$B = 256,$

$\lambda = 0.50,$

$t_R = 0.47$

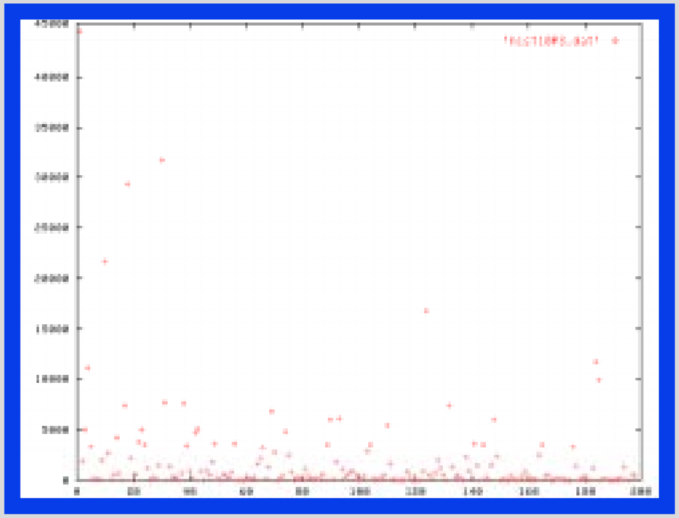
center

k-d Trees

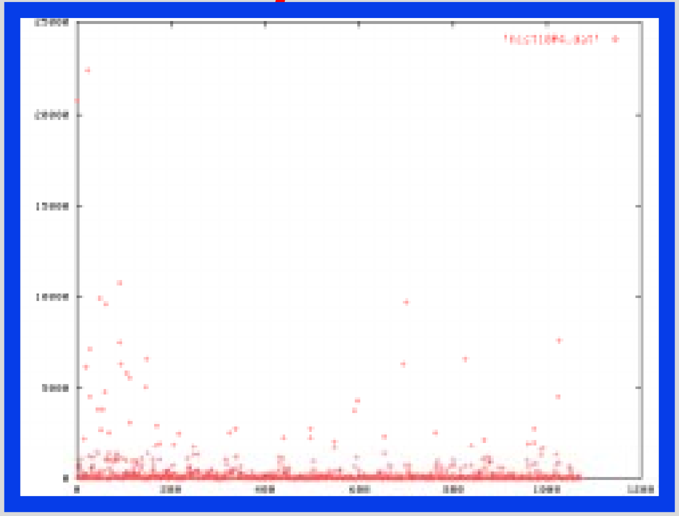
		RGB_8			RGB_5		
		M	t_B	t_Q	M	t_B	t_Q
Red-black	Max	1366	38.01	20.89	50	18.88	16.73
	Min	6	3.450	2.450	3	2.510	2.100
	Avg	334	22.39	15.44	17	13.95	13.09
Treap	Max	1366	45.85	24.27	50	21.44	17.03
	Min	6	3.310	2.300	3	2.330	2.070
	Avg	334	25.65	16.47	17	14.38	12.17

B=256, Largest range, center

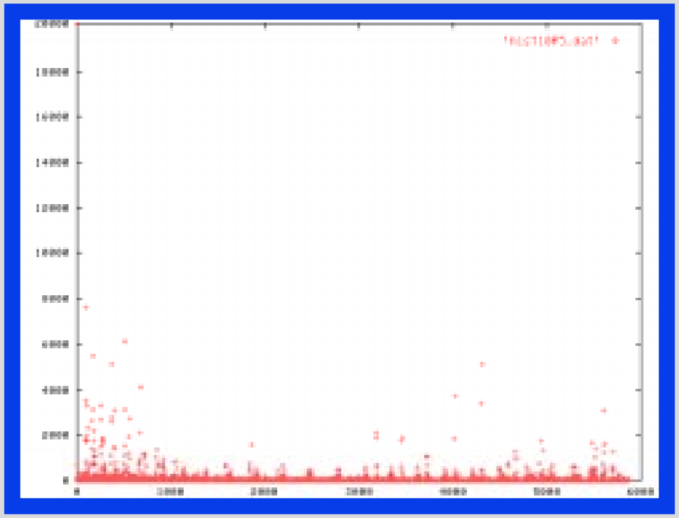
The Problem withTreaps



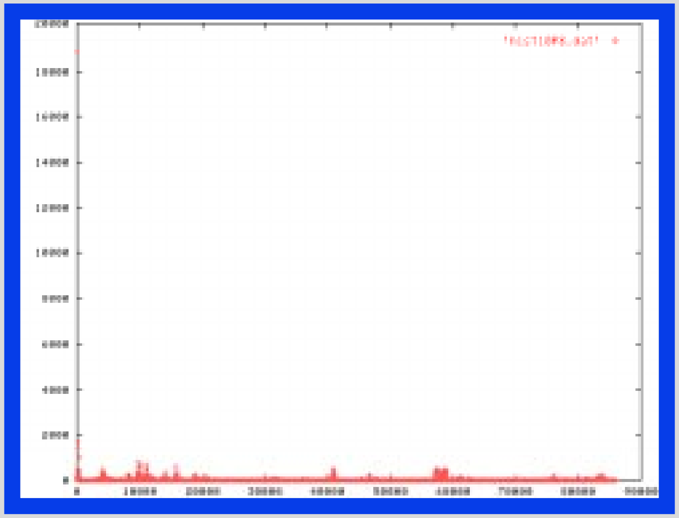
N_3 = 198; σ = 5,009



N_4 =1,070; σ = 1,368



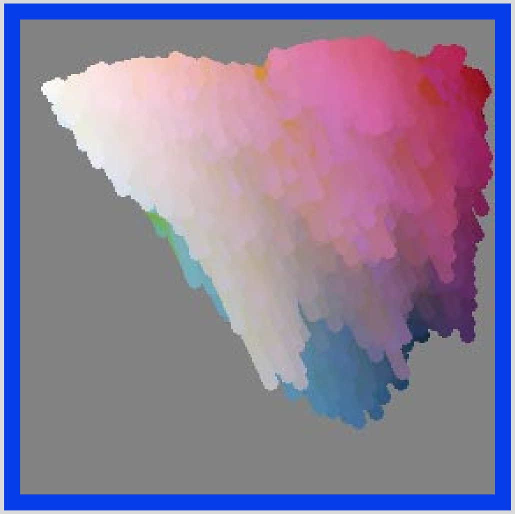
N_5 = 5,873; σ = 386



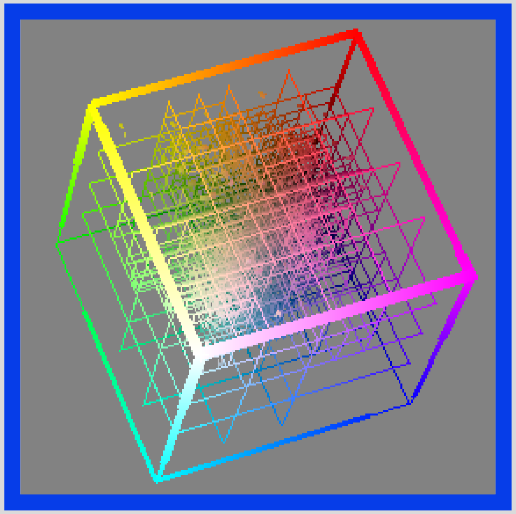
N_8 = 86,008; σ = 67

Summary

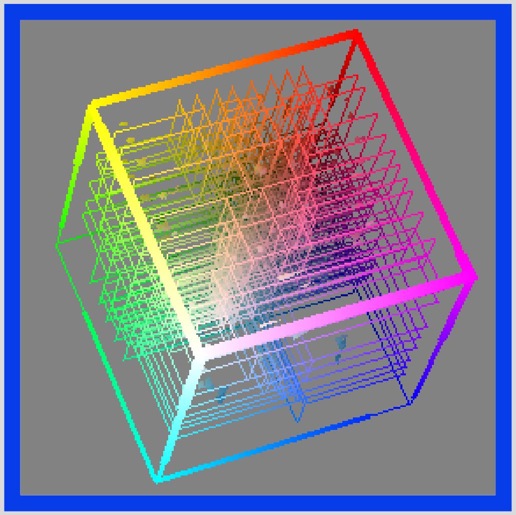
- Visual exploration of the space and time requirements of truecolor image histograms.
- Visualized the clustering property of spatial subdivisions.
- Red-black trees and treaps are competitive bucket structures, but red-black trees are best overall when $p \geq 7$ is desired.



2D Array



Octree



***k*-d tree**